Приложение к РПД

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

для проведения текущей и промежуточной аттестации по учебной

дисциплине

«Разработка программных приложений»

Для направления подготовки: 09.03.03 Прикладная информатика

Профиль: «Искусственный интеллект и прикладная информатика в цифровой экономике"

1. Описание показателей (дескрипторов) и критериев оценивания компетенций на различных этапах их формирования

Контроль качества освоения дисциплины включает в себя текущий контроль успеваемости и промежуточную аттестацию. Текущий контроль успеваемости и промежуточная аттестация обучающихся проводятся в целях установления соответствия достижений обучающихся поэтапным требованиям образовательной программы к результатам обучения и формирования компетенций.

Компетенции	Показатели (дескрипторы)	Критерии в соответствии с уровнем освоения ОП			Оценочное средство (промежуточная аттестация)
		пороговый (удовлетворительно)	стандартный (хорошо)	эталонный (отлично)	
ОПК-2	Знать	Имеет общее представление о - типологии, уровнях и особенностях языков высокого уровня, особенностях императивного и объектно - ориентированного программирования	Понимает типологию, уровни и особенностях языков высокого уровня, особенности императивного и объектно - ориентированного программирования	Имеет глубокие знания о типологии, уровнях и особенностях языков высокого уровня, особенностях императивного и объектно - ориентированного программирования	Теоретические вопросы
	Уметь	Умеет - выбирать средства программирования для решения поставленной задачи, - выбирать базовые технологии программирования, учитывая особенности среды разработки только под руководством преподавателя .	Умеет выбирать средства программирования для решения поставленной задачи, выбирать базовые технологии программирования, учитывая особенности среды разработки при незначительной помощи преподавателя.	Умеет полностью самостоятельно выбирать средства программирования для решения поставленной задачи, выбирать базовые технологии программирования, учитывая особенности среды разработки.	Теоретические вопросы
	Владеть	Владеет основами навыков выбора средств программирования для решения поставленной задачи, выбирать базовые технологии программирования, учитывая особенности среды разработки.	Владеет большей частью навыков выбора средств программирования для решения поставленной задачи, выбирать базовые технологии программирования, учитывая особенности среды разработки.	Владеет навыками выбора средств программирования для решения поставленной задачи, выбирать базовые технологии программирования, учитывая особенности среды разработки.	Практические задания

ОПК-7	Знать	Имеет представление и основы знаний о современных средствах автоматизации и программирования и возможностях современных языков программирования высокого уровня, источниках получения информации по особенностям современного программирования.	Имеет устойчивые знания о современных средствах автоматизации и программирования и возможностях современных языков программирования высокого уровня, источниках получения информации по особенностям современного программирования.	Имеет глубокие знания о современных средствах автоматизации и программирования и возможностях современных языков программирования высокого уровня, источниках получения информации по особенностям современного программирования.	Теоретические вопросы
	Уметь	Умеет создавать простые программы, учитывая особенность языка и среды разработки, - использовать источники получения информации по программированию только при помощи преподавателя.	Умеет создавать простые программы, учитывая особенность языка и среды разработки, - использовать источники получения информации по программированию в большинстве ситуаций самостоятельно.	Умеет создавать простые программы, учитывая особенность языка и среды разработки, - использовать источники получения информации по программированию полностью самостоятельно.	Практические задания
	Владеть	Владеет основами навыков создания простых программ, учитывая особенность языка и среды разработки, основами использования источников получения информации по программированию.	Владеет навыками создания простых программ, учитывая особенность языка и среды разработки, основами использования источников получения информации по программированию в большинстве ситуаций самостоятельно.	Владеет навыками создания простых программ, учитывая особенность языка и среды разработки, основами использования источников получения информации по программированию полностью самостоятельно.	Практические задания
ПК-2	Знать	Имеет представление и основы знаний: - об основных принципах разработки программного обеспечения, - о приёмах разработки требований, -об основах проектирования программного обеспечения.	Имеет устойчивые знания: - об основных принципах разработки программного обеспечения, - о приёмах разработки требований, -об основах проектирования программного обеспечения.	Имеет глубокие знания: - об основных принципах разработки программного обеспечения, - о приёмах разработки требований, -об основах проектирования программного обеспечения.	Теоретические вопросы

	Уметь	Умеет разрабатывать компоненты приложения в выбранной среде программирования,- настраивать свойства и программировать функционал основных компонентов приложения только при помощи преподавателя.	Умеет разрабатывать компоненты приложения в выбранной среде программирования,- настраивать свойства и программировать функционал основных компонентов приложения в большинстве ситуаций самостоятельно.	Умеет разрабатывать компоненты приложения в выбранной среде программирования,- настраивать свойства и программировать функционал основных компонентов приложения полностью самостоятельно.	Практические задания
	Владеть	Основами навыков разработки прикладного программного обеспечения на современных языках программирования, методами адаптации прикладного программного обеспечения.	Навыками разработки прикладного программного обеспечения на современных языках программирования, методами адаптации прикладного программного обеспечения в большинстве ситуаций самостоятельно.	Навыками разработки прикладного программного обеспечения на современных языках программирования, методами адаптации прикладного программного обеспечения полностью самостоятельно.	Практические задания
ПК-8	Знать	Демонстрирует знание особенностей тестирования программных приложений только при помощи преподавателя.	Демонстрирует знание особенностей тестирования программных приложений в большинстве ситуаций самостоятельно.	Демонстрирует знание особенностей тестирования программных приложений полностью самостоятельно.	Теоретические вопросы
	Уметь	Умеет формировать набор тестов, использовать современные системы автоматизированного тестирования только при помощи преподавателя.	Умеет формировать набор тестов, использовать современные системы автоматизированного тестирования в большинстве ситуаций самостоятельно.	Умеет формировать набор тестов, использовать современные системы автоматизированного тестирования полностью самостоятельно.	Практические задания
	Владеть	Демонстрирует владение навыками тестирования программных продуктов только при помощи преподавателя.	Демонстрирует владение навыками тестирования программных продуктов в большинстве ситуаций самостоятельно.	Демонстрирует владение навыками тестирования программных продуктов полностью самостоятельно.	Практические задания

2. Описание критериев и шкал оценивания результатов обучения по дисциплине (модулю)

2.1. Критерии и шкалы оценивания результатов обучения при проведении текущего контроля успеваемости

Текущий контроль предназначен для проверки хода и качества формирования компетенций, стимулирования учебной работы обучаемых и совершенствования методики

освоения новых знаний. Он обеспечивается проведением семинаров, оцениванием контрольных заданий, проверкой конспектов лекций, выполнением индивидуальных и творческих заданий, периодическим опросом обучающихся на занятиях. Контролируемые разделы (темы) дисциплины (модуля), компетенции и оценочные средства представлены в таблице.

№ п/п	Контролируемые разделы (темы) дисциплины*	Код контролируемой компетенции и/или индикаторы компетенции	Наименование оценочного средства**
1	Архитектура построения программного обеспечения	ОПК-2, 7(частично), ПК-2,8	Тестирование, устный блиц-опрос, индивидуальное практическое задание.
2	Проектирование и программирование комплекса программных средств	ОПК-2, 7, ПК-2,8	Тестирование, устный блиц-опрос, индивидуальное практическое задание.

Критерии и шкала оценивания результатов тестирования

Оценка	Критерий оценки
«отлично»	правильно выполненных заданий более 90%
«хорошо»	правильно выполненных заданий от 71 % до 89 %
«удовлетворительно»	правильно выполненных заданий от 50 % до 70 %
«неудовлетворительно»	правильно выполненных заданий меньше 50%

Критерии и шкала оценивания устных блиц-опросов

Оценка	Критерий оценки
«зачтено»	ответил верно на более чем 1/3 вопросов
«не зачтено»	правильные ответы составляют менее 1/3 части от ответов на все вопросы

Критерии и шкала оценивания индивидуального практического задания

Оценка	Критерий оценки
«зачтено»	все задачи решены верно, существенных замечаний по защите решений нет
«не зачтено»	правильно выполненных заданий меньше 50%, обучающийся не может пояснить решение большинства задач.

2.2. Критерии и шкалы оценивания результатов обучения при проведении промежуточной аттестации

Для оценивания результатов обучения при проведении промежуточной аттестации в 7-ом семестре используется 4-х балльная шкала.

Шкала оценивания	Критерии	Уровень освоения компетенций
Отлично	наличие глубоких и исчерпывающих знаний в объеме пройденного программного материала, правильные и уверенные действия по применению полученных знаний на практике, грамотное и логически стройное изложение материала при ответе, знание дополнительно рекомендованной литературы	Эталонный
Хорошо	наличие твердых и достаточно полных знаний программного материала, незначительные ошибки при освещении заданных вопросов, правильные действия по применению знаний на практике, четкое изложение материала	Стандартный
Удовлетвори тельно	наличие твердых знаний пройденного материала, изложение ответов с ошибками, уверенно исправляемыми после дополни-тельных вопросов, необходимость наводящих вопросов, правильные действия по применению знаний на практике	Пороговый
Неудовлетвор ительно	наличие грубых ошибок в ответе, непонимание сущности излагаемого вопроса, неумение применять знания на практике, неуверенность и неточность ответов на дополнительные и наводящие вопросы.	Компетенции не сформированы

Критерии и шкала оценивания курсовой работы

Критерии оценивания курсовых работ:

1. Соответствие теме

- Работа должна соответствовать заявленной теме проекта.
- Цель и задачи работы должны быть четко сформулированы и достигнуты.

2. Структура кода

- Код должен быть структурирован логично и последовательно.
- Использование функций, классов и модулей должно быть обоснованным.
- Четкое разделение функциональных блоков программы.

3. Читаемость и документирование кода

- Комментарии к коду должны быть информативными и полезными.
- Название переменных, функций и методов должно быть понятным и осмысленным.
- Наличие документации (например, docstrings).

4. Использование стандартных библиотек и сторонних пакетов

- Применение стандартных библиотек Python там, где это уместно.
- Правильное использование сторонних пакетов (если они необходимы), включая корректную установку и настройку.

5. Качество реализации алгоритмов

- Алгоритмы должны работать корректно и эффективно.
- Оптимизация кода по времени выполнения и использованию памяти.
- Отсутствие избыточного дублирования кода.

6. Тестирование и обработка ошибок

- Программа должна содержать тесты для проверки основных функций.
- Обработка исключений и непредвиденных ситуаций.
- Логика обработки ошибок должна быть прозрачной и безопасной.

7. Работа с данными
- Корректная работа с файлами, базами данных или другими источниками данных.
- Эффективность операций ввода-вывода.
- Безопасность при работе с внешними данными.

8. Интерфейс пользователя (GUI/CLI)
- Если проект предполагает наличие интерфейса, он должен быть интуитивно понятен и удобен для пользователя.
- Инструкции по использованию программы должны быть четкими и доступными.

9. Оформление работы
- Оформление пояснительной записки согласно требованиям учебного заведения.
- Соответствие оформления кода принятым стандартам PEP 8.
- Структурированная презентация результатов работы.

10. Самостоятельность выполнения
- Оценка оригинальности кода и отсутствие плагиата.
- Степень самостоятельности студента в выполнении задания.

11. Защита работы
- Умение аргументированно защищать свою работу перед преподавателем.
- Ответы на вопросы по коду и проекту.

Оценка	Критерий оценки
«отлично»	Не менее 90% критериев оценены на отлично
«хорошо»	Не менее 40% критериев оценены на хорошо, не менее 40% критериев оценены на отлично
«удовлетворительно»	Менее 50% критериев оценены на хорошо или отлично
«неудовлетворительно»	Более 50% критериев оценены на неудовлетворительно

3. Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы

3.1. Оценочные средства текущего контроля успеваемости

а) Примеры тестов

Раздел 1. Архитектура построения программного обеспечения

1. Что такое монолитная архитектура?
    a) Архитектура, где все компоненты приложения объединены в один исполняемый файл.
    b) Архитектура, где каждый компонент приложения работает независимо.
    c) Архитектура, где компоненты приложения взаимодействуют через API.
2. Что такое микросервисная архитектура?
    a) Архитектура, где все компоненты приложения объединены в один исполняемый файл.
    b) Архитектура, где каждый компонент приложения работает независимо.
    c) Архитектура, где компоненты приложения взаимодействуют через API.
3. Что такое API?
    a) Набор функций и процедур, предоставляемых приложением для использования другими программами.
    b) Набор инструкций для компилятора.
    c) Набор данных для анализа.
4. Что такое OASP?
    a) Открытая архитектура программного обеспечения.
    b) Операционная система.
    c) Оптимизация алгоритмов.
5. Что такое REST API?
    a) Архитектурный стиль, основанный на передаче состояния представления.
    b) Архитектурный стиль, основанный на удаленных вызовах процедур.
    c) Архитектурный стиль, основанный на обмене сообщениями.
6. Что такое RPC?
    a) Архитектурный стиль, основанный на передаче состояния представления.
    b) Архитектурный стиль, основанный на удаленных вызовах процедур.
    c) Архитектурный стиль, основанный на обмене сообщениями.
7. Что такое Message broker?
    a) Система, которая управляет обменом сообщениями между приложениями.
    b) Система, которая управляет удаленными вызовами процедур.
    c) Система, которая управляет передачей состояния представления.
8. Какая архитектура более масштабируема: монолитная или микросервисная?
    a) Монолитная.
    b) Микросервисная.
    c) Обе одинаково.
9. Какая архитектура проще в разработке: монолитная или микросервисная?
    a) Монолитная.
    b) Микросервисная.
    c) Обе одинаково.
10. Какая архитектура требует больше ресурсов для развертывания: монолитная или микросервисная?
    a) Монолитная.
    b) Микросервисная.
    c) Обе одинаково.

Ответы:

1. a) Архитектура, где все компоненты приложения объединены в один исполняемый файл.
2. b) Архитектура, где каждый компонент приложения работает независимо.

3. a) Набор функций и процедур, предоставляемых приложением для использования другими программами.
4. a) Открытая архитектура программного обеспечения.
5. a) Архитектурный стиль, основанный на передаче состояния представления.
6. b) Архитектурный стиль, основанный на удаленных вызовах процедур.
7. a) Система, которая управляет обменом сообщениями между приложениями.
8. b) Микросервисная.
9. a) Монолитная.
10. b) Микросервисная.

Раздел 2. Проектирование и программирование комплекса программных средств

1. Что такое жизненный цикл программного обеспечения?
    a) Процесс разработки программного обеспечения от идеи до завершения.
    b) Процесс тестирования программного обеспечения.
    c) Процесс внедрения программного обеспечения.
2. Что такое Agile?
    a) Методология разработки программного обеспечения, основанная на гибкости и итерациях.
    b) Методология разработки программного обеспечения, основанная на строгом планировании.
    c) Методология разработки программного обеспечения, основанная на документации.
3. Какие этапы включает жизненный цикл программного обеспечения?
    a) Анализ, проектирование, разработка, тестирование, внедрение, поддержка.
    b) Анализ, тестирование, внедрение.
    c) Проектирование, разработка, поддержка.
4. Что такое итерация в Agile?
    a) Повторяющийся цикл разработки, в котором создаются и тестируются новые функции.
    b) Процесс тестирования программного обеспечения.
    c) Процесс внедрения программного обеспечения.
5. Какие методы Agile наиболее популярны?
    a) Scrum, Kanban.
    b) Waterfall, V-model.
    c) XP, RUP.
6. Что такое Scrum?
    a) Методология Agile, основанная на спринтах и ежедневных встречах.
    b) Методология Agile, основанная на канбан-досках.
    c) Методология Agile, основанная на строгом планировании.
7. Что такое Kanban?
    a) Методология Agile, основанная на спринтах и ежедневных встречах.
    b) Методология Agile, основанная на канбан-досках.
    c) Методология Agile, основанная на строгом планировании.
8. Какие преимущества имеет Agile перед традиционными методологиями?
    a) Гибкость, быстрая адаптация к изменениям.
    b) Строгий контроль и планирование.
    c) Высокая степень документации.
9. Какие недостатки имеет Agile?
    a) Сложность в управлении большими проектами.
    b) Недостаток гибкости.

c) Высокие затраты на документацию.

10. Какие роли существуют в Scrum?
    a) Product Owner, Scrum Master, Development Team.
    b) Project Manager, Developer, Tester.
    c) Designer, Developer, QA Engineer.

Ответы:

1. a) Процесс разработки программного обеспечения от идеи до завершения.
2. a) Методология разработки программного обеспечения, основанная на гибкости и итерациях.
3. a) Анализ, проектирование, разработка, тестирование, внедрение, поддержка.
4. a) Повторяющийся цикл разработки, в котором создаются и тестируются новые функции.
5. a) Scrum, Kanban.
6. a) Методология Agile, основанная на спринтах и ежедневных встречах.
7. b) Методология Agile, основанная на канбан-досках.
8. a) Гибкость, быстрая адаптация к изменениям.
9. a) Сложность в управлении большими проектами.
10. a) Product Owner, Scrum Master, Development Team.

б) Примеры вопросов устных блиц-опросов

Раздел 1. Архитектура построения программного обеспечения

1. Что такое монолитная архитектура?
2. В чем преимущества монолитной архитектуры?
3. Какие недостатки у монолитной архитектуры?
4. Что такое микросервисная архитектура?
5. В чем преимущества микросервисной архитектуры?
6. Какие недостатки у микросервисной архитектуры?
7. Что такое API?
8. Какие типы API существуют?
9. Что такое REST API?
10. Что такое Message broker?

Раздел 2. Проектирование и программирование комплекса программных средств

1. Что такое жизненный цикл программного обеспечения?
2. Какие этапы включает жизненный цикл программного обеспечения?
3. Что такое Agile?
4. Какие принципы лежат в основе Agile?
5. Какие методы Agile наиболее популярны?
6. Что такое Scrum?
7. Какие роли существуют в Scrum?
8. Что такое Kanban?
9. Какие преимущества имеет Agile перед традиционными методологиями?
10. Какие недостатки имеет Agile?

в) Примеры индивидуальных практических заданий

Раздел 1. Архитектура построения программного обеспечения

1. Задание 1: Разработайте монолитное приложение на Python, которое включает в себя следующие компоненты: аутентификацию пользователей, управление пользователями и ведение журнала действий. Реализуйте взаимодействие между компонентами через внутренние вызовы функций.
2. Задание 2: Создайте микросервисное приложение, состоящее из трех сервисов: аутентификации, управления пользователями и ведения журнала действий. Реализуйте взаимодействие между сервисами через REST API. Используйте Docker для контейнеризации каждого сервиса.
3. Задание 3: Разработайте API на Python с использованием Flask или Django, который предоставляет следующие методы: регистрация пользователя, аутентификация пользователя, получение списка пользователей. Реализуйте аутентификацию через JWT (JSON Web Tokens).

Раздел 2. Проектирование и программирование комплекса программных средств

1. Задание 1: Разработайте проектную документацию для приложения, которое управляет задачами сотрудников. Включите в документацию описание требований, архитектуры, дизайна и плана тестирования. Используйте методологию Agile для планирования и управления проектом.
2. Задание 2: Создайте прототип приложения для управления задачами сотрудников, используя методологию Scrum. Определите роли в команде (Product Owner, Scrum Master, Development Team), создайте бэклог продукта и проведите спринт-планирование.
3. Задание 3: Разработайте приложение для управления задачами сотрудников, используя методологию Kanban. Создайте канбан-доску для отслеживания задач, определите этапы работы (To Do, In Progress, Done) и проведите итерацию разработки.

3.2. Оценочные средства промежуточной аттестации

Перечень теоретических вопросов (для оценки знаний):

7 семестр

1. Что такое монолитная архитектура и в каких случаях она предпочтительна?
2. Какие преимущества и недостатки имеет монолитная архитектура?
3. Что такое микросервисная архитектура и в каких случаях она предпочтительна?
4. Какие преимущества и недостатки имеет микросервисная архитектура?
5. Что такое API и каковы его основные функции?
6. Какие типы API существуют и в чем их различия?
7. Что такое REST API и какие принципы лежат в его основе?
8. Что такое RPC и как он отличается от REST API?
9. Что такое Message broker и как он используется в архитектуре программного обеспечения?
10. Какие примеры Message broker существуют и как они применяются?
11. Что такое жизненный цикл программного обеспечения и какие этапы он включает?

12. Какие методы проектирования программного обеспечения существуют и в чем их различия?
13. Что такое Agile и какие принципы лежат в его основе?
14. Какие методы Agile наиболее популярны и в чем их особенности?
15. Что такое Scrum и как он применяется в разработке программного обеспечения?
16. Какие роли существуют в Scrum и какие задачи они выполняют?
17. Что такое Kanban и как он применяется в разработке программного обеспечения?
18. Какие преимущества имеет Agile перед традиционными методологиями разработки?
19. Какие недостатки имеет Agile и в каких случаях его применение может быть затруднено?
20. Каковы основные этапы проектирования программного обеспечения и какие инструменты используются на каждом этапе?

Перечень типовых задач (для оценки умений):

1. Задача 1: Разработайте монолитное приложение на Python, которое включает в себя следующие компоненты: аутентификацию пользователей, управление пользователями и ведение журнала действий. Реализуйте взаимодействие между компонентами через внутренние вызовы функций.
2. Задача 2: Создайте микросервисное приложение, состоящее из трех сервисов: аутентификации, управления пользователями и ведения журнала действий. Реализуйте взаимодействие между сервисами через REST API. Используйте Docker для контейнеризации каждого сервиса.
3. Задача 3: Разработайте API на Python с использованием Flask или Django, который предоставляет следующие методы: регистрация пользователя, аутентификация пользователя, получение списка пользователей. Реализуйте аутентификацию через JWT (JSON Web Tokens).
4. Задача 4: Разработайте проектную документацию для приложения, которое управляет задачами сотрудников. Включите в документацию описание требований, архитектуры, дизайна и плана тестирования. Используйте методологию Agile для планирования и управления проектом.
5. Задача 5: Создайте прототип приложения для управления задачами сотрудников, используя методологию Scrum. Определите роли в команде (Product Owner, Scrum Master, Development Team), создайте бэклог продукта и проведите спринт-планирование.
6. Задача 6: Разработайте приложение для управления задачами сотрудников, используя методологию Kanban. Создайте канбан-доску для отслеживания задач, определите этапы работы (To Do, In Progress, Done) и проведите итерацию разработки.

4. Методические материалы, определяющие процедуру оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

4.1. Описание процедур проведения текущего контроля успеваемости студентов

В таблице представлено описание процедур проведения контрольно-оценочных мероприятий текущего контроля успеваемости студентов, в соответствии с рабочей программой дисциплины, и процедур оценивания результатов обучения с помощью спланированных оценочных средств.

Наименование оценочного средства	Описания процедуры проведения контрольно-оценочного мероприятия и процедуры оценивания результатов обучения
Тестирование	*Очная форма обучения.* Тестирование проводится по результатам освоения разделов дисциплины во время практических занятий. Во время проведения тестирования пользоваться учебниками, справочниками, конспектами лекций, тетрадями для практических занятий не разрешено. Преподаватель на практическом занятии, предшествующем занятию проведения теста, доводит до обучающихся: темы, количество заданий в тесте время выполнения. . На одном из практических занятий во время сессии проводится комплексное тестирование по всем темам курса.
Устный блиц-опрос	*Очная форма обучения.* Опрос проводится фронтально на каждом занятии в течении 3-5 минут с целью актуализации опорных знаний и проверки готовности к изучению следующей темы. Опросы не проводятся.
Индивидуальное практическое задание	*Очная форма обучения.* Индивидуальные практические задания выдаются на практических занятиях, последующих после изучения темы на лекции. Индивидуальные задания должны быть выполнены в установленный преподавателем срок и в соответствии с требованиями Выполненные задания в назначенный срок проверяются путём демонстрации работы программы в среде программирования и отчёта по результатам тестирования программы. Индивидуальные практические задания выполняются в рамках контрольной работы, которая сдаётся на проверку до сессии и защищается на практических занятиях.

4.2. Описание процедур проведения промежуточной аттестации

Экзамен

Экзамен проводится в традиционной форме, по билетам.

При определении уровня достижений обучающих на экзамене обращается особое внимание на следующее:

- дан полный, развернутый ответ на теоретический вопрос;
- практические задания (задачи) решены верно;

- показана совокупность осознанных знаний по основам программирования, проявляющаяся в свободном оперировании понятиями, умении применять алгоритмические конструкции и типы данных для решения задач;
- ответ формулируется в терминах дисциплины, изложен литературным языком, логичен, доказателен, демонстрирует понимание программирования.

Пример экзаменационного билета:

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ Федеральное государственное бюджетное образовательное учреждение высшего образования «Забайкальский государственный университет»	**ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 12** по дисциплине **Разработка программных приложений** направление подготовки 09.03.03 Прикладная информатика 2 семестр 20 / * уч. года

1. Объектно-ориентированное программирование: причины возникновения, достоинства и недостатка, абстрагирование, объекты и классы, понятия: событие, метод. Виды алгоритмических конструкций.
2. Практическое задание № 12

СОСТАВИЛ: кафедры ПИМ _____ « » 20 г.	УТВЕРЖДАЮ: Зав. кафедрой ПИМ *_____ « » 20 г.

Курсовая работа

Методические указания по написанию курсовой работы по дисциплине "Разработка программных приложений"

Введение

Курсовой проект по дисциплине "Разработка программных приложений" направлен на закрепление теоретических знаний и практических навыков, полученных в ходе изучения курса. В рамках курсовой работы студенты должны продемонстрировать умение проектировать и разрабатывать программное обеспечение, используя различные архитектурные подходы и методологии.

Структура курсовой работы

1. Введение
    - Обоснование выбора темы.
    - Цели и задачи работы.
    - Актуальность темы.
2. Теоретическая часть
    - Описание выбранной архитектуры (монолитная или микросервисная).

- Анализ существующих API (REST, RPC, Message broker).
- Обзор методологий разработки (Agile, Scrum, Kanban).

3. Практическая часть
    - Проектирование системы.
    - Реализация программного обеспечения.
    - Тестирование и отладка.
4. Заключение
    - Выводы по работе.
    - Рекомендации по дальнейшему развитию проекта.
5. Список литературы
    - Перечень использованных источников.

Требования к курсовой работе

1. Объем работы: 20-30 страниц.
2. Оформление: Согласно ГОСТу (шрифт Times New Roman, 14 pt, интервал 1.5).
3. Графические материалы: Диаграммы, схемы, скриншоты.
4. Программный код: Приложения с пояснениями.

Этапы выполнения курсовой работы

1. Выбор темы и постановка задачи
    - Консультация с преподавателем.
    - Определение целей и задач работы.
2. Теоретическое исследование
    - Изучение литературы по выбранной теме.
    - Анализ существующих решений.
3. Проектирование системы
    - Разработка архитектуры.
    - Проектирование API.
    - Выбор методологии разработки.
4. Реализация программного обеспечения
    - Написание программного кода.
    - Интеграция компонентов.
5. Тестирование и отладка
    - Проведение тестирования.
    - Исправление ошибок.
6. Написание отчета
    - Оформление курсовой работы.
    - Подготовка к защите.

Критерии оценки

1. Теоретическая часть: Глубина анализа, обоснованность выбора архитектуры и методологии.
2. Практическая часть: Качество реализации, функциональность, соответствие требованиям.
3. Отчет: Структура, оформление, ясность изложения.
4. Защита: Уверенность в ответах, умение аргументировать свою позицию.

Примерные темы курсовых работ

1. Разработка монолитного приложения для управления задачами с использованием REST API.
2. Проектирование и реализация микросервисного приложения для интернет-магазина с использованием Message broker.
3. Сравнение монолитной и микросервисной архитектур на примере разработки системы управления проектами.
4. Разработка API для системы аутентификации пользователей с использованием JWT (JSON Web Tokens).
5. Проектирование и реализация системы управления задачами с использованием Agile-методологии.
6. Разработка REST API для системы управления контентом (CMS).
7. Проектирование и реализация системы мониторинга серверов с использованием RPC.
8. Сравнение REST API и RPC на примере разработки системы управления пользователями.
9. Разработка системы управления задачами с использованием Kanban-методологии.
10. Проектирование и реализация системы управления складом с использованием микросервисной архитектуры.
11. Разработка API для системы управления заказами с использованием OASP.
12. Проектирование и реализация системы управления проектами с использованием Scrum-методологии.
13. Сравнение Agile и Waterfall методологий на примере разработки системы управления задачами.
14. Разработка системы управления задачами с использованием монолитной архитектуры и REST API.
15. Проектирование и реализация системы управления контентом с использованием микросервисной архитектуры и Message broker.